REMARKS

The Examiner rejected claims 1-36 as obvious (35 U.S.C. §103) over "Introduction to Algorithms", by Cormen et al., ISBN 0-07-013143-0 (referred to herein as "Cormen") and "Data Structures and Other Objects Using", by Main et al., ISBN 0-8053-7470-1 (referred to herein as "Main". Applicants traverse for the following reasons.

Independent claims 1, 10, and 19 concern processing an input file in a file system, wherein the input file has an input file name. A function is applied to map the input file name to a value. A data structure is processed to determine whether there is a preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps, wherein two files that map to a same value according to the function are capable of having a same name.

The Examiner cited Cormen as teaching a hash function in general and a map with a hash table slot to which a hash table value maps. (Office Action, p. 3) However, nowhere does the cited Cormen anywhere suggest applying a function to map an input file name to a value and processing a data structure to determine whether there is a preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps. Instead, the cited Cormen just discusses hash functions in general, not as applied to a file system as claimed.

The Examiner cited pages 545-546 and 548 of Main as teaching the claim requirements concerning using the function to map an input file name to a value and processing the data structure to determine whether the file system has a name that maps to the same value as the input file name. (Office Action, pg. 3)

The cited pages 545-546 discuss a hash function applied to a field in a record, such as a stock number of a tractor record, to hash the stock number value to a fewer number of possible hash values to conserve the amount of space that needs to be provided in the record to store the track number.

Nowhere do these cited pages of Main anywhere teach or suggest applying a function to an input file name in a file system and then processing a data structure to determine whether

there is a preexisting file system having a name that maps, according to the function, to the same value to which the input file name maps. Instead, the cited pages 545-546 only discuss applying a hash function to a field in a record, not an input file to determine whether there is a preexisting file in the file system having a same name.

The Examiner cited lines 23-28 of page 548 which discuss an is_present function that searches the hash table for a record with a particular key. If the hash table has such a key, then a find function is called to return a copy of the record with that key. This cited section nowhere teaches or suggests applying a hash function to an input file name as claimed and then processing a data structure to determine whether the file system has a preexisting file having a name that maps, according to the function, to the same name as the input file.

Because the cited Main and Cormen nowhere teach or suggest applying a hash function to a file system as claimed, the Examiner is effectively modifying the cited Main to apply the hashing technique of Main to a file system to determine whether a file system has a preexisting file whose name maps, according to the function, to a same name as an input file.

Applicants submit that the proposed modification of Main is inappropriate because the Examiner has not cited any art that teaches or suggests applying a function, such as a hash function, to map an input file name to a value as part of determining whether a file system has a preexisting file whose name maps to the input file name.

The Examiner further cites the file system discussed in the Background Section of the Application ("Background Section"). However, the Background Section nowhere suggests the claimed technique for determining whether a preexisting file exists in the file system using a function and data structure as claimed. In fact, the Background Section discusses the prior art technique that searches the entire file system for a matching preexisting file, over which the claimed invention is intended to be an improvement.

The Manual of Patent Examination Procedure (MPEP) states that the "mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination." MPEP 2143.01, pg. 2100-124. Here, the Examiner's proposed modification of the cited art is improper because the

Examiner has not cited any art that suggests applying a function, such as a hash function, to file names in a file system to determine whether an input file name has a same name as a preexisting file as claimed.

In fact, by combining the hash function of Cormen with Main, all one has is the hash function of Cormen applied to determining whether a record is present having a stock number value that hashes to the hash of a desired stock number value. There is no suggestion anywhere in the cited art that would apply the hash functions of Cormen and Main to a file system as claimed, such as the file system discussed in the Background Section.

The Examiner further found that the modification would be obvious because it would optimize searching the storage space of a file system. (Office action, pg. 4). Applicants traverse this finding because nowhere does any cited art teach or suggest applying a hash function to file names to improve searching of file names in a file system. This purported motivation is one use of the claimed invention and nowhere taught or suggested in any cited art. Thus, the Examiner is engaging in inappropriate use of hindsight by citing a use of the claimed invention discussed in the Application (see, pgs, 3-4 and 8-9) to justify the proposed modification because the cited motivation is nowhere taught or suggested in the cited prior art.

For all these reasons, Applicants submit that claims 1, 10, and 19 are patentable over the combination of Cormen and Main.

Claims 2-9 and 28-30; 11-18 and 31-33; and 20-27 and 34-36 are patentable over the cited combination of Cormen and Main because they depend from claims 1, 10, and 19, respectively, which are patentable over the cited combination for the reasons described above. Further claims discussed below provide additional grounds of patentability over the cited art.

Claims 2, 11, and 20 depend from claims 1, 10, and 19, respectively, and further require that the mapped-to values require fewer bits of storage than the file names. The Examiner cited page 222 of Cormen as disclosing the additional requirements of these claims. (Office Action, pg. 4) Applicants traverse.

The cited Cormen only discusses hash functions in general and nowhere suggests that a function is applied to file names to map file names in a file system to values requiring fewer bits of storage than the file name.

Accordingly, claims 2, 11, and 20 provide additional grounds of patentability over the cited art because the additional requirements of these claims are nowhere taught or suggested in the cited art, alone or in combination.

Claims 4, 13, and 22 depend from claims 3, 12, and 13, which require that the data structure includes an entry for each possible integer value capable of being generated from the hash function. Claims 4, 13, and 22 further require that processing the data structure to determine whether there is a preexisting file comprises determining whether the entry for the integer value to which the input file name maps indicates the presence of one preexisting file mapping to the same integer value as the input file name.

The Examiner cited pg. 547 of Main as teaching the additional requirements of these claims. (Office Action, pg. 5) Applicants traverse.

The cited pg. 547 discusses how to avoid collisions when determining a hash key for the key value of the record, such as the stock number, and then storing the record in a hash key entry that is vacant.

Claims 4, 13, and 22 require that an entry in the data structure to which the input file name maps indicates the presence of one preexisting file mapping to the same value as the input file name. The cited Main 547 discusses determining whether a hash array already has a record at a location corresponding to the hash value (data[hash(key)]) and if not, storing the record at the location corresponding to the hash value (data[hash(key)]) and if so storing the record at a next free location (data[hash(key)+1]).

Nowhere does the cited Main anywhere suggest processing the data structure do determine whether the entry in the data structure corresponding to the value of the input name indicates the presence of one preexisting file in the file system mapping to the same integer value as the input file name. Instead, the cited Main only discusses how the hash array is processed to associate key values with a hash value in a manner that avoids collisions. Nowhere does the

cited Main teach or suggest using the values of a data structure, such as a hash array, to determine whether an input file name has a same name as a preexisting file in the file system as claimed.

For these reasons, claims 4, 13, and 22 provide additional grounds of patentability over the cited combined art.

Claims 5, 14, and 23 depend from claims 4, 13, and 22 and further require that the data structure is a one-dimensional array and wherein each entry is capable of having one of two values. The entry is set to a first value if there is one preexisting file name in the file system that maps to the integer value for the entry. Determining whether there is one preexisting file comprises determining whether the entry for the integer value to which the input file name maps has the first value.

The Examiner cited page 222 of Cormen as teaching the additional requirements of these claims. (Office Action, pgs. 5-6) The cited page 222 discusses a hash array having hash table slots to which hash values map.

Nowhere does the cited Cormen anywhere teach or suggest the claim requirement that each entry in the array maps to one of two values that indicates whether there is or is not a preexisting file name in the file system that maps to the integer value corresponding to the entry.

The Examiner found that the hash table of Cormen either has a hash value or is null. However, the hash table of Cormen teaches away from the claim requirement because each entry in the hash table of Cormen may have a different value corresponding to the different possible hash values. The claims on the other hand require that each entry only have one of two values. The cited hash table of Cormen teaches away from this requirement because the entries can have either a null value or any one of the possible numerous hash values, not one of two values as claimed.

Further, nowhere does the cited Cormen teach or suggest that the hash table entries may have a same first value indicating the presence of a preexisting file as claimed.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 5, 14, and 23.

Claims 6, 15, and 24 depend from claims 1, 10, and 19 and further require applying the function to each file name in the file system to map each file name to one value and indicating in the data structure, for each file name, that there is one preexisting file for the value to which the file name maps.

The Examiner cited page 222 of Cormen as teaching the additional requirements of these claims. (Office Action, pgs. 6-7) The Examiner cited Cormen discussion of applying the hash function to the possible key values. The Examiner found that it would be obvious to apply Cormen's teaching of hash keys corresponding to actual keys to applying the hash function to each file name in a file system. Applicants traverse.

Nowhere does the cited Cormen, or any other cited art, anywhere teach or suggest applying a function to each file name in a file system and indicating in a data structure for each file name that there is one preexisting file for the value to which the file name maps. Instead, the cited Cormen only discusses how a hash function may map keys to hash table slots. There is no suggestion anywhere of applying a hash, or any other function, to the file names and indicating in the hash table entries whether a file name corresponding to the entry exists.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 6, 15, and 24.

Claims 7, 16, and 25 depend from claims 6, 15, and 24 and further require that the input file is the subject of an access request. Further, each file in the file system is scanned to determine if there is at least one preexisting file having the same name as the input file name if there is one preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps.

The Examiner found that it would be obvious to modify Cormen and Main to apply to the file system prior art discussed in the Background Section of the Application to teach the claim requirements. (Office Action, pgs. 7-8) Applicants traverse.

The cited Background Section discusses prior art techniques for determining whether a file exists in a file system by scanning the entire file system for a matching file name. However, nowhere is there any suggestion in the cited Background Section of applying a function to an

input file name to determine whether the input file name value maps to a preexisting file name as claimed.

Indeed, nowhere does the Background Section, or any other cited art, anywhere teach or suggest scanning each file in the file system on condition that there is at least one preexisting file having a name that maps, according to the function, to the same value to which the input file name subject to the access request. Instead, the Background Section only discusses scanning all files without regard to whether the input file name maps to a same value as a preexisting file. Further, nowhere does the cited Cormen or Main anywhere teach or suggest applying a hash function to a file name subject to an access request to determine whether there is a preexisting file whose name hashes to the same value as the input file.

Again, the Examiner is modifying prior art in a manner nowhere taught or suggested in the cited art. The proposed modifications are based solely on the claim requirements and not suggested in any cited art. Although different elements of the claims may be separately discussed in the cited art, such as a file system, hash functions, etc., nowhere is there any art that suggests the combination of these requirements as claimed to determine whether there is a preexisting file for a requested file in a file system.

Accordingly, the cited references do not teach or suggest, alone or in combination, the requirements of claims 7, 16, and 25.

Claims 8, 9, 17, 18, 26, and 27 depend from claims 7, 16 or 14 and provide further details on file system access operations based on using the claimed technique to determine whether there is a preexisting file having the same name as the input file subject to the access request. Because the Examiner has cited the same art in rejecting these claims (Office Action, pgs. 7-8), the cited combination fails to disclose the additional requirements of these claims in combination with the base and intervening claims from which they depend.

Claims 28, 31, and 34 depend from claims 1, 10, and 19 and further require searching the file system for one file having the same name as the input file name if the data structure indicates that one preexisting file has a name that maps, according to the function, to the same value to

which the input file maps. An operation is performed if the file system includes one file having the same name as the input file.

The Examiner again cited the Background Section of the Application as teaching searching a file system. (Office Action, pgs. 9-10) However, nowhere does the cited Background Section anywhere suggest searching a file system for a preexisting file name if the data structure indicates that one preexisting file name maps to the same value as the input file name. Further, nowhere does the cited Main anywhere suggest using a data structure and function as claimed to determine whether there is a preexisting file with a same name as the input file. Again, the Examiner is modifying the cited art in manners nowhere taught or suggested in the cited art. The Examiner's proposed modifications are improper because the only motivation or suggestion for these proposed modifications is found in the claims themselves and the disclosure of the Application.

Applicants further submit that claims 29, 30, 32, 33, 35, and 36 are also patentable over the cited art because they depend from claims 28, 31, and 34, which are patentable over the cited art for the reasons discussed above, and because the additional requirements concerning the file system operations are nowhere taught or suggested in the cited prior art in combination with the claimed function and data structure used to determine whether an input file has a same name as a preexisting file.

Conclusion

For all the above reasons, Applicant submits that the pending claims 1-36 are patentable over the art of record. Applicants submit that no fees are needed. Nonetheless, should any additional fees be required, please charge Deposit Account No. 50-0585.

The attorney of record invites the Examiner to contact him at (310) 553-7977 if the Examiner believes such contact would advance the prosecution of the case.

Dated: September 3, 2002

By: _____
David W. Victor
Reg. No.: 39,867

Please direct all correspondences to:

David Victor
Konrad Raynes Victor & Mann, LLP
315 South Beverly Drive, Ste. 210
Beverly Hills, CA 90212
Tel: 310-553-7977
Fax: 310-556-7984